A Java-based Multi-Institutional Medical Information Retrieval System

Kang Wang Ph.D.*, drs F.J. van Wingerde*, Karen Bradshaw BS*,
Peter Szolovits Ph.D.+, Isaac Kohane MD Ph.D.*
*Boston Children's Hospital Informatics Program
+Laboratory for Computer Science, MIT

Abstract

JAMI (Java-based Agglutination of Medical Information) is designed as a framework for integrating heterogeneous information systems used in healthcare related institutions. It is one of the implementations under the W3-EMRS project [1][i] aimed at using the World Wide Web (Web) to unify different hospital information systems.

JAMI inherited several design decisions from the first W3-EMRS implementation described in [2], including using the Web as the communication infrastructure and HL7 as the communication protocol between the heterogeneous systems and the W3-EMRS systems. In addition, JAMI incorporates the growing Java technologies and has a more flexible and efficient architecture. This paper describes JAMI's architecture and implementation. It also present two instances of JAMI, one for the integration of different hospital information systems and another for the integration of two heterogeneous systems within a single hospital. Some important issues for the further development of JAMI, including security and confidentiality, data input and decision support are discussed.

Motivations

In recent years, cost pressure from managed care has been causing consolidation of healthcare institutions. As a result, the integration of independently developed information systems becomes one of the foci of both the medical informatics research community and the information system providers. Even inside a single hospital, different departments usually have different vendor specific information systems. Demands for integrating these systems are also increasing.

Various integration strategies exist. One is to create open interfaces among different vendor specific systems, often based on the exchange of data via the HL7 [3] protocol, and often with the intent of feeding all data to a new data repository. Although this is a welcome advance over previous systems that could not communicate at all, the resulting systems still suffer from many typical disadvantages. Most of the systems being integrated and most data repository solutions remain platform dependent, locking the organization into required support for old hardware as well as new for the repository. Some systems take advantage of client-server architecture, but often the internal communication protocols are proprietary and thus hinder further integration. In such systems, HL7 is used only for communication with external systems, and makes it difficult to enhance functionality without turning again to the vendor who supplied the proprietary protocols. Another popular strategy is to create new data warehouses that either duplicate or supplant existing legacy systems. However, capturing the full capabilities of the older systems often involves a long and complex analysis task to determine all the legacy data models. Also, solutions that duplicate data require solutions to difficult data synchronization problems. The warehousing strategy therefore usually leads to many years' development efforts and high costs.

A sustainable integration system for healthcare institutions must be open and scaleable. Therefore it should be based on standard and widely supported protocols. To reduce maintenance costs, the systems should be easy to install and to update. The system should also be platform independent in order to make use of an institution's past investment on hardware and software.

JAMI tries to reach these goals by using Web as the communication infrastructure and HL7 as the communication protocol both between the underlying legacy systems and JAMI and between the components of JAMI. Furthermore, JAMI uses Java as the implementation language and the component infrastructure JavaBean as the framework. Therefore, JAMI is easily adaptable to different institutions by adding, changing or configuring its components. Also, it is possible to use standard Java-enabled Web browsers as JAMI's front end, which makes JAMI easy to access. The use of Web browsers enables JAMI to access volumes of medical resources on the Internet such as MEDLINE and to take advantages of the formatting capabilities of the HyperText Markup Language (HTML).

In the next section, some Web-based medical record systems are reviewed, including our first HTML-based implementation in the W3-EMRS project. Some fundamental limitations of these systems are presented. In Section 3, the architecture of JAMI is described. We discuss how Java technology enables us to overcome the limitations outlined in Section 2. Two modules of the architecture, the Common Medical Record and the User Interface, are discussed in more detail in Section 4. The advantages of component-oriented system development are illustrated. In Section 5, two instances of JAMI are described. They are used to integrate multiple hospital information systems and multiple information systems within a

---

[i] Visit http://www.emrs.org/ for more information and demonstrations

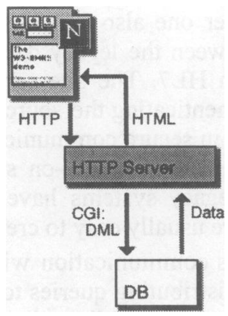

Figure 1: First Generation Web-Based Systems

hospital respectively. Finally, some interesting issues that arose during the development of JAMI as well as planned future efforts are discussed in Section 6.

Previous work

In its infancy, the Web was mostly used for accessing static information presented in the form of HTML pages. Today, some systems combine Web and database technologies to generate Web pages on the fly. To use a Web browser as the front end, a hospital data repository as the back-end and CGI programs[i] as the mediator to retrieve (store) information from (to) the repository characterized the first generation of systems that used a Web front-end to medical records [4, 5]. Usually, the first generation systems only dealt with single databases. The typical architecture of these systems is shown in Figure 1.

The Web browser sends a query (e.g. get the last 24 hours' lab results) to the Web server, which in turn start a CGI program to handle the query. The CGI program translates the query into the data manipulation language (DML, e.g. SQL) of the underlying repository and retrieves the data. It then represents the data in HTML and sends it back to the browser.

Previous HTML-based implementations in the W3-EMRS project expanded the first generation Web front ends to handle information retrieved from multiple hospital information systems [2, 6]. The architecture has an abstract layer, called the Common Medical Record (CMR), above the underlying legacy information models. For each legacy system, a *site server* must be written which is responsible for the translation between the legacy information model and the CMR. Figure 2 shows the architecture of the system.

The Web browser sends queries in the form of HL7 messages to a Web server (e.g. get the last 5 visits for this patient). The Web server in turn invokes a CGI program (called the *agglutinator*) which distributes the query to the participating site servers and gathers the returned CMR encoded in HL7. Since the implementation assumes that the browser only understands HTML, the agglutinator must do all the merg-

[i] http://hoohoo.ncsa.uiuc.edu/cgi/

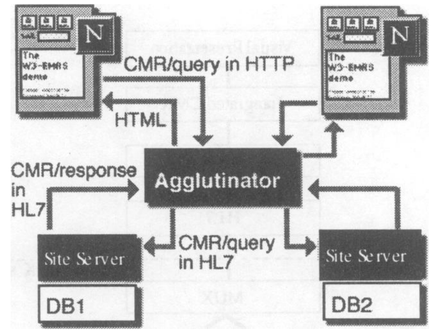

Figure 2: HTML-based W3-EMRS

ing and translation of the CMR (e.g. vocabulary translation of the problem list, merging the clinical notes from different sites and sorting them by date). At the end, the agglutinator translates the result into HTML and sends it back to the browser.

There are some fundamental limitations brought by the assumption that the Web browsers only understand HTML. First, HTML has very limited data types, namely text and images; furthermore, HTML supplies limited layout and formatting operations on these data types (e.g. paragraph, table, list). It is difficult, for example, to use these data types and operations to present a live EKG-wave for an ICU application. Although the functionality of some Web browsers (e.g. Netscape) can be expanded by using extra programs (e.g. plug-ins) to handle new data types, these extra programs are browser dependent and usually platform dependent as well.

Second, the user interaction model of HTML is very simple. Essentially, the only user interaction is to go to a certain place in a specified HTML file, which might be generated on the fly. Even though this simplicity contributed largely to the popularity of Web, it is a severe limitation for systems whose underlying domains, such as medicine, require complex logic and user interaction. This problem has been widely recognized. Now, many web browsers allow scripts to be embedded into HTML which define more sophisticated user interaction. But again these scripting languages are vendor specific.

Third, HTML only represents the appearance (view) of the data but not the data-model itself. For example, if we want to use HTML to present a trend of a given lab parameter, the image data type must be used. But if an abnormal point is noticed in the trend and the user wants to know the value behind the point, the browser must go back to the Web server to get the value because the set of values behind the trend, which constitutes the model of the view, is not represented by HTML.

Fourth, since HTML can only be used to describe the view of the underlying data model, the server must do all the translations from HL7 to CMR to HTML. The resulting HTML file, together with the related image files, might use much more space than it is

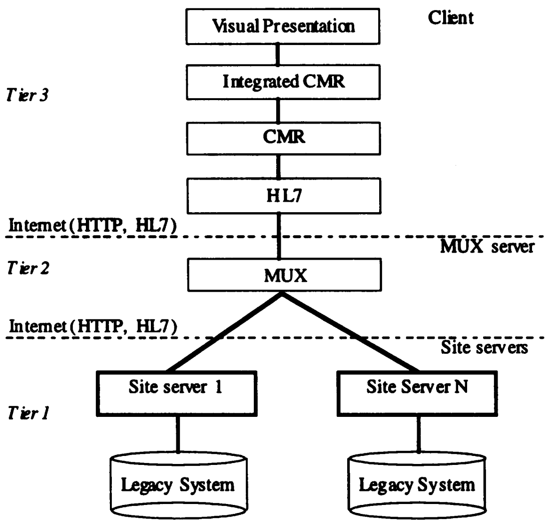

Figure 3: Architecture of JAMI

necessary to store the data model itself. This makes the HTML-based implementation demanding both on the server's computation power and the network's speed.

Last but not least, the communication protocol of Web, HTTP, is stateless. As a result, all the information (called state) that is needed for generating a Web page (e.g. patient ID, user name and password for login to the database) must be passed to the corresponding CGI-program. Even if a Web page itself might not need the state, it still needs to store it (usually in forms of hidden fields) in order to be able to pass it to the linked pages, whose generation needs the state. To keep and pass these states among large numbers of Web-pages is a huge overhead, especially when the states are complex. There are extensions of HTTP which tried to remedy this flaw. For example, cookies are introduced to remember some simple states. But it is hard to use them to represent complex states.

Due to the rapid development of the Web technologies, it is now reasonable to assume that Web-browsers not only understand HTML but they can also run Java-applets. This new assumption leads to the new design of the W3-EMRS system as described in the next section. The new design overcomes the above mentioned limitations, which restricted the design of the HTML-based system of W3-EMRS.

Architecture

The design of JAMI inherited several ideas from the previous HTML-based design of W3-EMRS. These include using Web as the communication infrastructure and HL7 as the communication protocol. JAMI also reused the CMR defined as part of the HTML-based system and the Agglutinator functionality. JAMI's three tiered architecture is shown in Figure 3.

Tier one contains the legacy data repositories. These are mostly existing hospital or departmental information systems. Tier one also contains the site servers that translate between the legacy data model and the CMR encoded in HL7. The site servers are also responsible for authenticating the source of the request, and participating in secure communication by encryption and enforcement of agreed-on security policies. Because many legacy systems have HL7 interfaces, the site servers are usually easy to create.

Tier two supports communication with multiple legacy systems by distributing queries to appropriate tier 1 site servers and assembling their individual responses into a combined response returned to the requester. MUX, or *multiplexor*, refers to combining multiple data streams into one. Both requests and responses are encoded in HL7, and the security architecture enhances reliability and confidentiality of communications. This tier is also concerned with noting unavailability of any of the information sources and strategies for finding alternative communication paths in case of network difficulties. Because Tier 2 may be implemented on machines distinct from those supporting the other tiers, this architecture can improve performance by distributing loads. Compared with the HTML-based design, the load of tier two may be reduced by distributing a large part of the computation to the third tier. The load of the network is also reduced because the patient record can be encoded more concisely in HL7 than in HTML.

The third tier is loaded as a Java applet to a Web-browser running on the client. The HL7 module decodes the HL7 messages into segments and fields, upon which the CMR is built. The "Integrated CMR" module integrates the CMR from different legacy systems according to the semantics of the records. For example, clinical notes from different hospitals can be assembled into a chronological sequence, and vocabulary translations allow assembly of a common list of prescribed medications or a joint list of diagnostic conclusions. The integrated CMR is presented by the Visual Presentation module.

Because HL7 is used as the communication protocol both between MUX and the client-side applet and between MUX and the site servers. it is possible for the site servers to be MUXs, or interface engines, that manage their own site servers. In this way, hierarchies can be built to reflect the organizational structure of the to be integrated institutions.

One of the design decisions made for JAMI as shown in Figure 3 is that the computational tasks such as data format translation, vocabulary are accomplished as much as possible by the client, which are typically a single user PCs or workstations. However, the line between client and server can be moved. For example, if we have a powerful server hardware and weak client machines, the division line can be set between Agglutination and Visual Presentation (VP). With Java's Remote Method Invocation mechanism, it is relatively easy to implement this shift.

In the current implementation, all tiers except tier one have been implemented in JAVA. Tier one is closely related to the legacy systems. An implementation language should be chosen according to the underlying hardware platform and data repository.

It should be pointed out that most modules in the architecture need to be adapted to the concrete integration situations. In the next section, we detail the adaptation process of two modules.

Common Medical Record and Visual Presentation

The structure of the CMR depends largely on the data model of the underlying legacy systems and the target users of the integrated system. However, experience showed that there are relatively constant elements of the CMR. With a repository of such elements, it is easy to construct a CMR for a new case.

In the first phase of the W3-EMRS project, the Boston Collaborative defined a CMR which contains: Patient Demography, Visits History, Clinical Notes, Diagnosis/Problems, Medications, Allergies. Each element has attributes which were mapped to all the HIS of the participating hospitals.

Another module which is highly dependent on the concrete integration systems is the Visual Presentation module. The way the CMR should be displayed and the way the users interact with the system depend, among others, on the user group (e.g. physicians, nurses, administrators), where the system operates (e.g. Emergency Room, ICU) and the usage of the system (e.g. documentation, monitoring, decision support). Similar to the CMR module, visualization elements can be identified which can be reused in different integration systems.

In the implementation level, component methodology is used to map the elements in the CMR module and Visual Presentation module to software components. Concretely, Java Bean[i] is used in JAMI as the component infrastructure.

Application examples

JAMI is being tested and evaluated for the integration of the hospital information systems (HIS) of Boston Children's Hospital, Beth-Israel Hospital and Mass General Hospital in the Boston area [6]. JAMI also is used to integrate a vendor-specific ICU system with the HIS at Children's Hospital. Other related efforts include using W3-EMRS to integrate the HIS of Beth-Israel and Deaconess Hospitals to meet the new information systems requirements imposed by the merger of these two institutions [9]. Additionally, W3-EMRS is being used to merge the medical records of new-born babies with jaundice at Brigham & Women's Hospital, Beth Israel Hospital and Children's Hospital, and to implement guidelines based on the data obtained from the merged records.

In the following, we describe in more detail two instances of JAMI. The first is used to integrate multiple disparate HIS. The second is used to integrate an ICU data repository with a HIS within a hospital.

Multiple Hospitals

This instance is designed for the doctors in emergency rooms (ER). It helps ER doctors to save their precious time by releasing them from login to different systems for retrieving patient record. After login to the JAMI system, the doctors input some basic patient demographic information and then choose which parts of the patient record need to be retrieved (Figure 4). Requests are sent automatically to the participating sites. Note that no effort is made to solve the master patient index (MPI) problem. Instead, simple matching of the basic demographics fields is used. Based on the returned, more comprehensive, patient demographics data (Figure 5) the clinicians decide whether the responding sites have relevant information for their patient. An MPI system can be plugged into the site servers to implement more flexible matching strategies.

After the patient medical record is retrieved, the VP module presents the data to the clinicians. Figure 6 presents the problem list of the patient from two hospitals. Note that the screen Figure 6 is composed of two components, one for displaying the demographics and one for displaying the problem list.

Multiple Data Repositories within a Hospital

This instance is designed for the physicians and nurses in intensive care units (ICU). Many ICUs have their own vendor specific data repositories with high data density. Web front-end has been developed for integrating a specific ICU system and a specific HIS [7]. This instance of JAMI re-implements the system described in [7]. It makes full use of Java and is more efficient and flexible. The use of JAMI architecture makes it easier to be expanded and to be ported.

A user chooses a patient from a patient list that is automatically updated by the ADT system of the underlying ICU application. Several screens are defined that display different facets of the patient record. Figure 7 is part of the "current status" screen. It shows the most important vital signal waves and the latest BLOODGAS lab result. The lab data and wave data come from different data sources, but the difference is transparent to the user.

Discussions

Data input

The current implementation of JAMI is focused on the viewer functionality. However, since HL7 is a bidirectional communication protocol, there is no limitation that excludes data input. In fact, by using Java at the client site, input of complex data structures (e.g. order entry) with complex validation logic can be realized.

[i] http://splash.javasoft.com/beans/

One obstacle of implementing data input is the management policy of the sites. It is generally more difficult to get the permission to write to a data repository than to read from it. One of the possible solutions is to implement an extra site which stores user input such as annotations, notes, etc.

Security and Confidentiality

Security and confidentiality is a general concern of using the Internet, especially when sensitive data such as patient records are transmitted. The W3-EMRS project defined a comprehensive security and confidentiality framework that includes authentication of site servers, multiplexors and client sites, user access based on the combination of designated password and one time password and encrypted transmission of messages. The most important component of the framework is a coherent security and confidentiality policy among the participating institutions [8].

Decision Support and Guidelines

Having access to the complete historical medical record of the patient lets clinical applications like report makers, automated guidelines help deliver better care by informing clinicians to a previously unattainable level. One of the current projects at Boston Children's Hospital is to create an automated guideline for urgent care of jaundiced newborns. Implementation of this system on top of the W3-EMRS architecture allows us to retrieve and incorporate birth and maternal data from the neighboring birth-hospitals for consideration when recommending a course of action. Such a system has the potential to shorten the length of time to search for the patient data and thus shorten the time between the child being admitted and the start of effective and appropriate treatment.

Acknowledgments

The project W3-EMRS has been supported by NLM, SpaceLabs Medical, Sun Microsystems and Oracle.

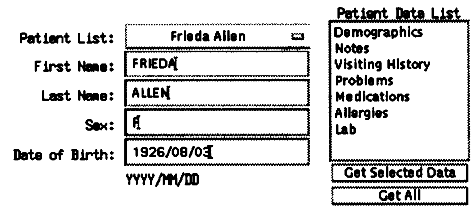

Figure 4: Patient Identification

Site(s)	chibos-ears	bi-ears
Given Name	FRIEDA	FRIEDA
Family Name	ALLEN	ALLEN
Date of Birth	8/03/26	8/03/26
Sex	F	F
Address	3 Mass. Ave. Boston 02157 MA	3 Mass. Ave. Boston 02157 MA
Home Number	8005551211	6175551212

Figure 5: Information from multiple HIS

FRIEDA ALLEN 1926/08/03 F

Problems

Source	Date Est.	Name
bi-ears	11/13/92	HEMATURIA
	7/29/92	HTN
	7/29/92	HYPERCHOLESTEROLEMIA
	7/29/92	HYPOTHYROIDISM
	7/29/92	RIGH PTOSIS SECONDARY TO THYROID SURGERY
	7/29/92	DIVERTICULITIS
	7/29/92	RENAL COLIC
	7/29/92	TB IN PAST
chibos-ears	6/30/34	PRIMARY-HYPOTHYROIDISM
	6/30/34	THYROID-DISEASE
	6/13/34	OBESITY

Figure 6: Problem List

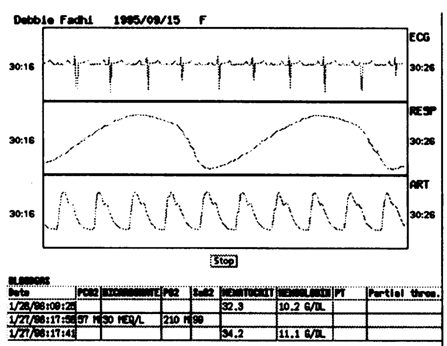

Figure 7: Integration of ICU and HIS data

References

1. Kohane IS, Greenspun P, Fackler J, Cimino C, Szolovits P. Building National Electronic Medical Record Systems via the World Wide Web. *JAMIA* 1996;3:191-207.

2. F.J. van Wingerde et al. Using HL7 and the World Wide Web for Unifying Patient Data from Remote Databases. In: Cimino JJ, ed. *1996 AMIA Annual Fall Symposium*. Washington, DC, USA: Hanley & Belfus, Inc; 1996:643-647.

3. Health Level Seven: An application protocol for electronic data exchange in healthcare environments; Version 2.2,*Health Level Seven*. Chicago, Illinois; 1990.

4. Cimino JJ, Socratous SA, Clayton PD. Internet as Clinical Information System: Application development using the World Wide Web. *JAMIA*. 1995;2:273-284.

5. Willard KE, Hallgren JH, Sielaff B, Connelly DP. The deployment of a World Wide Web (W3) based medical information system. In: Gardner RM, ed. *Symposium on Computer Applications in Medical Care*. New Orleans, Louisiana: Hanley & Belfus, Inc; 1995:771-775.

6. Isaac Kohane et al. Sharing Electronic Medical Records Across Multiple Heterogeneous and Competing Institutions. In: Cimino JJ, ed. *1996 AMIA Annual Fall Symposium*. Washington, DC, USA: Hanley & Belfus, Inc; 1996:608-612.

7. Kang Wang, Isaac Kohane, Karen Bradshaw, James Fackler. A Real Time Patient Monitoring System on the World Wide Web. In: Cimino JJ, ed. *1996 AMIA Annual Fall Symposium*. Washington, DC, USA: Hanley & Belfus, Inc; 1996:729-732.

8. Rind DM, Kohane IS, Szolovits P, Safran C, Chueh HC, Barnett GO., Maintaining the Confidentiality of Medical Records Shared over the Internet and World Wide Web, Accepted for publication in: Annals in Internal Medicine 1997

9. Halamka J, Safran C, Virtual Integration of the Beth Israel and Deaconess Hospitals via the WWW, Procedings of the 1997 AMIA Fall Symposium